CLAIMS

What is claimed is:

1. A method for automatically collecting garbage objects, comprising:

performing root set enumeration in a stack of a thread using a stack trace cache;

obtaining a root set by combining references obtained from root set enumeration for all threads;

tracing live objects that are reachable from references in the root set; and

reclaiming storage space occupied by objects other than the live objects.

2. The method of claim 1, wherein the root set comprises currently live references in stacks, registers, and other storage areas for all threads.

3. The method of claim 1, wherein performing root set enumeration in a stack of a thread using a stack trace cache comprises:

determining if a stack frame is cached in the stack trace cache;

detecting a portion of stack trace information that has not changed since the last stack enumeration in the stack trace cache and retrieving the unchanged portion from the stack trace cache, if the stack frame is cached in the stack trace cache; and

performing normal root set enumeration in the stack, if the stack frame is not cached in the stack trace cache.

4. The method of claim 1, wherein tracing the live objects comprises marking the live objects and further scanning objects that are reachable from the marked objects.

5. A method for performing root set enumeration in a stack of a thread for automatic garbage collection, comprising:

initializing a stack enumeration list;

creating a stack trace cache for the thread; and

performing root set enumeration in the stack using the stack trace cache to produce an updated stack enumeration list.

6. The method of claim 5, further comprising tagging a stack frame to indicate whether the stack frame has been cached in the stack trace cache or is newly generated in the stack and has not been cached in the stack trace cache.

7. The method of claim 5, wherein the stack enumeration list comprises currently live references in the stack.

8. The method of claim 5, wherein creating the stack trace cache comprises:

determining if the root set enumeration in the stack is performed for the first time;

dynamically creating the stack trace cache and caching information of stack frames in the stack trace cache, if the root set enumeration in the stack is performed for the first time; and otherwise,

reusing the stack trace cache for the root set enumeration in the stack.

9. The method of claim 8, wherein caching information of stack frames comprises adding an identifier to each frame using values of an instruction pointer register and a stack pointer register.

10. The method of claim 8, wherein reusing the stack trace cache comprises at least one of the following: using the stack trace cache without a change, and modifying the stack trace cache to accommodate new and updated stack frames.

11. The method of claim 5, wherein performing root set enumeration in the stack using the stack trace cache comprises:

determining if a stack frame is cached in the stack trace cache based on a tag of the stack frame;

copying an unchanged portion of stack trace information from the stack trace cache to the stack enumeration list, if the value of the tag indicates that the stack frame is cached in the stack trace cache; and otherwise

performing normal root set enumeration in stack.

12. The method of claim 11, wherein copying the unchanged portion of the stack trace information comprises detecting a portion of the stack trace information that has not changed since the last stack enumeration in the stack trace cache and retrieving the unchanged portion from the stack trace cache.

13. The method of claim 11, wherein performing normal stack enumeration comprises:

unwinding the stack until reaching a frame that is cached in the stack trace cache;

caching information of unwound frames in the stack trace cache; and

adding enumerated references from the unwound frames into the stack enumeration list.

14. A managed runtime system, comprising:

a virtual machine;

a just-in-time compiler;

a root set enumeration mechanism capable of obtaining a root set;

at least one stack trace cache capable of storing trace information of stack frames when the stack frames are enumerated during root set enumeration; and

a garbage collector capable of tracing live objects reachable from the root set and reclaiming storage space occupied by objects other than the live objects.

15. The system of claim 14, wherein the root set enumeration mechanism comprises a stack enumeration mechanism capable of performing root set enumeration in stack.

16. The system of claim 14, wherein the stack trace cache comprises an identification area to store stack frame identifiers and a list area to store enumerated references for stack frames.

17. A system for root set enumeration in a stack during garbage collection, comprising:

a stack frame classifier capable of classifying a stack frame based on a tag of the stack frame;

a stack information caching mechanism capable of caching information of the stack frame in a stack trace cache, if the value of the tag indicates that the stack frame is not cached;

an unchanged trace detecting mechanism capable of detecting a portion of stack trace information that has not changed since the last stack enumeration in the stack trace cache, if the value of the tag indicates that the stack frame is cached; and

an unchanged trace retrieving mechanism capable of retrieving the unchanged portion of stack trace information from the stack trace cache.

18. The system of claim 17, further comprising a stack unwinding mechanism capable of unwinding a stack until reaching a stack frame that is cached in the stack trace cache.

19. The system of claim 17, wherein the trace information caching mechanism comprises:

a cache creator capable of creating the stack trace cache when root set enumeration in the stack is performed for the first time; and

a frame identification component capable of identifying a frame in the stack trace cache.

20. An article comprising: a machine accessible medium having content stored thereon, wherein when the content is accessed by a processor, the content provides for automatically collecting garbage objects by:

performing root set enumeration in a stack of a thread using a stack trace cache;

obtaining a root set by combining references obtained from root set enumeration for all threads;

tracing live objects that are reachable from references in the root set; and

reclaiming storage space occupied by objects other than the live objects.

21. The article of claim 20, wherein the root set comprises currently live references in stacks, registers, and other storage areas for all threads.

22. The article of claim 20, wherein content for performing root set enumeration in a stack of a thread using a stack trace cache comprises content for:

determining if a stack frame is cached in the stack trace cache;

detecting a portion of stack trace information that has not changed since the last stack enumeration in the stack trace cache and retrieving the unchanged portion from the stack trace cache, if the stack frame is cached in the stack trace cache; and

performing normal root set enumeration in the stack, if the stack frame is not cached in the stack trace cache.

23. The article of claim 20, wherein content for tracing the live objects comprises content for marking the live objects and further scanning objects that are reachable from the marked objects.

24. An article comprising: a machine accessible medium having content stored thereon, wherein when the content is accessed by a processor, the content provides for performing root set enumeration in a stack of a thread for automatic garbage collection by:

initializing a stack enumeration list;

creating a stack trace cache for the thread; and

performing root set enumeration in the stack using the stack trace cache to produce an updated stack enumeration list.

25. The article of claim 24, further comprising content for tagging a stack frame to indicate whether the stack frame has been cached in the stack trace cache or is newly generated in the stack and has not been cached in the stack trace cache.

26. The article of claim 24, wherein the stack enumeration list comprises currently live references in the stack.

27. The article of claim 24, wherein content for creating the stack trace cache comprises content for:

determining if the root set enumeration in the stack is performed for the first time;

dynamically creating the stack trace cache and caching information of stack frames in the stack trace cache, if the root set enumeration in the stack is performed for the first time; and otherwise,

reusing the stack trace cache for the root set enumeration in the stack.

28. The article of claim 27, wherein content for caching information of stack frames comprises content for adding an identifier to each frame using values of an instruction pointer register and a stack pointer register.

29. The article of claim 27, wherein content for reusing the stack trace cache comprises at least one of the following: content for using the stack trace cache without a change, and content for modifying the stack trace cache to accommodate new and updated stack frames.

30. The article of claim 24, wherein content for performing root set enumeration in the stack using the stack trace cache comprises content for:

determining if a stack frame is cached in the stack trace cache based on a tag of the stack frame;

copying an unchanged portion of stack trace information from the stack trace cache to the stack enumeration list, if the value of the tag indicates that the stack frame is cached in the stack trace cache; and otherwise

performing normal root set enumeration in stack.

31. The article of claim 30, wherein content for copying the unchanged portion of the stack trace information comprises content for detecting a portion of the stack trace information that has not changed since the last stack enumeration in the stack trace cache and content for retrieving the unchanged portion from the stack trace cache.

32. The article of claim 30, wherein content for performing normal stack enumeration comprises content for:

unwinding the stack until reaching a frame that is cached in the stack trace cache;

caching information of unwound frames in the stack trace cache; and

adding enumerated references from the unwound frames into the stack enumeration list.